Evaluación

Inserta tu texto aquí.

*Obligatorio

1. Nombre *

____

1.-En el programa siguiente, dada la estructura struct alumno, indicar qué sentencias se han de sustituir por el COMENTARIO en la función leer_datos para que se lea el nombre y la nota de un alumno, si dicha función recibe como argumento un puntero a dicha estructura.

```
struct alumno
{
  char *nombre;
  int nota;
};
void ingresardatos(struct alumno *);
int main(void)
{
  struct alumno al;
  ingresardatos(&al);
  return 0;
}
void ingresardatos(struct alumno *alu)
{
  char nom[80];
  puts("Nombre: ");
  gets(nom);
  /* COMENTARIO */
  puts("Nota: ");
  scanf("%d",&alu->nota);
}
```

1. strcpy(alu->nombre,nom);

2. gets(alu->nombre);

3.
```
 alu->nombre=(char *)malloc((strlen(nom)+1));
 if (alu->nombre==NULL)
 {
   puts("No hay memoria");
   return;
 }
 alu->nombre=nom);
```

4.
```
 alu->nombre=(char *)malloc((strlen(nom)+1));
 if (alu->nombre==NULL)
 {
   puts("No hay memoria");
   return;
 }
 strcpy(alu->nombre,nom);
```

2. 1.- Elegir una opción * 1 punto

Marca solo un óvalo.

- Opción 1
- Opción 2
- Opción 3
- Opción 4

2.-Con las siguientes declaraciones, indique qué instrucción puede ser correcta:

```
struct ficha
{
char nom[40];
int edad;
};
struct ficha *pun=NULL;
```

3. 2.- Elegir una opción 1 punto

Marca solo un óvalo.

- pun=(struct ficha *)malloc(4*sizeof(struct ficha));
- pun=(struct ficha *)calloc(3,sizeof(struct ficha *));
- pun=(struct ficha *)realloc(pun,5*sizeof(40*char));
- pun=(struct ficha *)malloc(sizeof(struct ficha *));
- Otros: ____________________

4. 3.- En una pila se han ido almacenando las siguientes informaciones por orden de llegada: MA,JA,CO,GR,SE,BU, siendo MA la información que lleva más tiempo en la pila. Si se hacen dos operaciones de sacar y una operación de poner en la que se pone el dato AL. * 1 punto

Marca solo un óvalo.

- Con la siguiente operación sacar obtendremos AL y con la siguiente obtendremos BU.
- Con la siguiente operación sacar obtendremos AL.
- Con la siguiente operación sacar obtendremos CO.
- Con la siguiente operación sacar obtendremos BU y con la siguiente obtendremos SE.

5. 4.-En una pila se han ido almacenando los siguientes números por orden de llegada:27, 33, 12, 20, 8, 10, siendo 27 el dato que lleva más tiempo en la pila. Si se hacen dos operaciones de sacar y una operación de poner en la que se mete el dato 15. * 1 punto

Marca solo un óvalo.

- Con la siguiente operación sacar obtendremos 20.
- Con la siguiente operación sacar obtendremos 15 y con la siguiente obtendremos 20.
- Con la siguiente operación sacar obtendremos 12.
- Con la siguiente operación sacar obtendremos 12 y con la siguiente obtendremos 20.

6. 5.-En una cola se han ido almacenando los siguientes números por orden de llegada:27, 33, 12, 20, 8, 10, siendo 27 el dato que lleva más tiempo en la cola. Si se hacen dos operaciones de sacar y una operación de poner en la que se mete el dato 15. * 1 punto

Marca solo un óvalo.

- Con la siguiente operación sacar obtendremos 12 y con la siguiente pop obtendremos 20.
- Con la siguiente operación sacar obtendremos 15.
- Con la siguiente operación sacar obtendremos 15 y con la siguiente pop obtendremos 20.
- Con la siguiente operación sacar obtendremos 33.

6.-Una lista enlazada está formada por estructuras con el siguiente formato:

```
struct pp
{
  int num;
  struct pp *sig;
};
```

Se suponen además las siguientes declaraciones:

```
struct pp *p,*q;
```

Indique qué instrucciones son necesarias para borrar el elemento siguiente al elemento apuntado por el puntero p, suponiendo que sí existe tal elemento siguiente.

1.
```
p->sig=(tipo *)malloc(sizeof(tipo));
p->sig->sig=p->sig;
free(p);
```

2.
```
q=p->sig;
p->sig=q;
free(p);
```

3.
```
q=p->sig;
p->sig=q->sig;
free(q);
```

4.
```
q=p->sig;
*p=*q;
free(q);
```

7. 6.- Elegir una opción *

1 punto

Marca solo un óvalo.

- Opción 1
- Opción 2
- Opción 3
- Opción 4

7.-La función fun busca un elemento en una lista enlazada, en la que cada elemento tiene el siguiente tipo de datos:

```
struct nodo
{
  int dato;
  struct nodo *sig;
};
```

El primer parámetro p es un puntero al primer elemento de la lista y el segundo parámetro num es el número que se busca. La función fun debe devolver la dirección del elemento de la lista, si encuentra el numero a buscar, y devuelve NULL si no se encuentra el número en la lista. Indicar qué sentencias hay que sustituir por el comentario.

```
struct nodo *fun(struct nodo *p, int num)
{
  int sw=0;
  /* COMENTARIO A SUSTITUIR */
}
```

1.
```
  while((p != NULL) && (sw == 0))
    if( p->dato==num)
      sw=1;
    else
      p++;
  if (sw)
    return(p);
  else
    return NULL;
```

2.
```
  while((p->sig != NULL) && (sw == 0))
    if( p->dato==num)
      sw=1;
    else
      p=p->sig;
  if (sw)
    return(p);
  else
    return NULL;
```

3.
```
  while((p != NULL) && (sw == 0))
    if( p->dato==num)
      sw=1;
    else
      p=p->sig;
  if (sw)
    return(p);
  else
    return NULL;
```

4.
```
  while((p != NULL) && (sw == 0))
    if( p->dato==num)
      sw=1;
    else
      p=p->sig;
  if (sw)
    return(--p);
  else
```

```
return NULL;
```

8. 7.- Elegir una opción * 1 punto

Marca solo un óvalo.

- Opción 1
- Opción 2
- Opción 3
- Opción 4

8.-Una lista enlazada está compuesta por estructuras del siguiente tipo:

```
struct pp
{
  int num;
  struct pp *sig;
};
```

Indique la instrucción que falta para que la función recorrer() recorra e imprima la lista enlazada cuya dirección de comienzo se le da como argumento de entrada.

```
void recorrer(struct pp *pun)
{
  while(pun!=NULL)
  {
    printf("%d ", pun->num);
    /*  Instrucción que falta      */
  }
}
```

9. 8.- Elegir una opción * 1 punto

Marca solo un óvalo.

- pun++;
- pun=pun->sig;
- pun=pun+1;
- pun->sig=pun;

9.-Se ha realizado una lista telefónica por medio de una lista enlazada formada por estructuras con el siguiente formato:

```
struct pp
{
  char nom[20]; // nombre
  unsigned int tel; // numero de telefono
  struct pp *sig;
};
```

Indique cómo podría ser la función buscar(), a la que se le pasa como argumentos la dirección de comienzo p de la lista enlazada y un nombre nom, y devolverá como resultado el teléfono correspondiente a dicho nombre. Si el nombre a buscar no se encuentra en la lista, devolverá 0.

1.
```
unsigned buscar(struct pp *p,char *nom)
{
  unsigned tel=0;
  while (p=NULL)
  {
    if (strcmp(nom,p->nom)==0)
      return p->tel;
    p=p->sig;
  }
  return tel;
}
```

2.
```
unsigned *buscar(struct pp *p,char *nom)
{
  unsigned tel=0;
  while (p!=NULL)
  {
    if (strcmp(nom,p->nom)==0)
      return p->tel;
    p=p->sig;
  }
  return *tel;
}
```

3.
```
unsigned buscar(struct pp *p,char *nom)
{
  unsigned tel=0;
  while (p->sig!=NULL)
  {
    if (strcmp(nom,p->nom)>0)
      return p->tel;
    p=p->sig;
  }
  if (strcmp(nom,p->nom)==0)
    return p->tel;
  return tel;
}
```

4.
```
unsigned buscar(struct pp *p,char *nom)
{
  unsigned tel=0;
  while (p!=NULL)
  {
    if (strcmp(&nom[0],&p->nom[0])==0)
      return p->tel;
```

```
    p=p->sig;
  }
  return tel;
}
```

10. 9.- Elegir una opción * 1 punto

Marca solo un óvalo.

- Opción 1
- Opción 2
- Opción 3
- Opción 4

10.-En una lista simplemente enlazada formada por elementos del tipo struct elem y apuntada por un puntero global lista que apunta al primer elemento de la lista, indicar cuál es la función insertar adecuada para insertar un nuevo elemento al final de la lista, incluso si la lista inicial está vacía.

```
struct elem
{
  int dato1;
  char dato2;
  struct elem *siguiente
};

struct elem *lista;
```

1.

```
void insertar(int d1, char d2)
{
  struct elem *p = lista, *q = (struct elem *) malloc(sizeof(struct elem));
  if (q!=NULL)
  {
    q->dato1=d1;
    q->dato2=d2;
    q->siguiente=NULL;
    while (p->siguiente!=NULL)
      p = p->siguiente;
    p->siguiente = q;
  }
}
```

2.

```
void insertar(int d1, char d2)
{
  struct elem *p = lista, *q = (struct elem *)malloc(sizeof(struct elem));
  if (q==NULL)
    return;
  q->dato1=d1;
  q->dato2=d2;
  q->siguiente=NULL;
  if (p==NULL)
  {
    lista=q;
    return; }
  while (p->siguiente!=NULL)
    p = p->siguiente;
  p->siguiente = q;
}
```

3.

```
void insertar(int d1, char d2)
{
  struct elem *p = lista, *q = (struct elem *)malloc(sizeof(struct elem));
  if (q==NULL)
    return;
  q->dato1=d1;
  q->dato2=d2;
  q->siguiente=NULL;
  if (p==NULL)
    lista=q;
  else
  {
    while (p!=NULL)
      p = p->siguiente;
```

```
    p->siguiente = q;
  }
}

4.
void insertar(int d1, char d2)
{
  struct elem *p = lista, *q;
  if ((q = (struct elem *)malloc(sizeof(struct elem)))==NULL)
    return;
  q->dato1=d1;
  q->dato2=d2;
  q->siguiente=NULL;
  if (p==NULL)
  {
    lista=q;
    return;
  }
  while (p->siguiente!=NULL)
    p = p->siguiente;
  p->siguiente = lista;
}
```

11. 10.- Elegir una opción * 1 punto

Marca solo un óvalo.

- Opción 1
- Opción 2
- Opción 3
- Opción 4